We’re pleased to announce blastula, a package for creating beautiful custom  
emails in R. At RStudio, we love interactive dashboards, but some situations  
call for a different communication mechanism. Use blastula to:

* **Compose** custom email bodies based on code, code output, and markdown
* **Send** emails using SMTP servers – even GMail – or integrate with production services like RStudio Connect

Blastula makes it easy to send notifications for everything from anomaly detection to fantasy basketball results, all without leaving R.

To get started, install blastula from CRAN:

install.packages('blastula')

**Creating Email**

Blastula’s unique strength is creating custom HTML email bodies that render in a  
variety of email clients, including mobile. The recommended way to create email  
is using blastula’s R Markdown output format. blastula::blastula\_email. The  
body of the email will respect the R Markdown output, including markdown syntax  
and code chunk outputs.

---

output: blastula::blastula\_email

---

Hi Team,

This \*important\* forecast needs to go out today.

```{r echo=FALSE}

model <- arima(presidents, c(1, 0, 0))

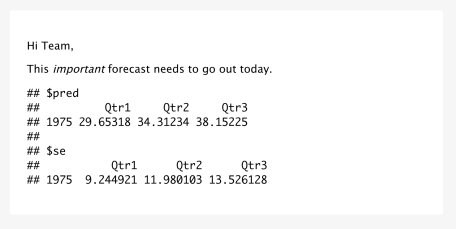
predict(model, 3)

```

To create the email from the R Markdown document, use render\_email:

email <- render\_email('email.Rmd')

The resulting email object can be previewed in RStudio.



Alternatively, it is possible to create an email without R Markdown, by using  
the compose\_email function to combine text, images, and even plots:

library(blastula)

library(ggplot2)

library(glue)

plot <- qplot(disp, hp, data = mtcars, colour = mpg)

plot\_email <- add\_ggplot(plot)

email <- compose\_email(

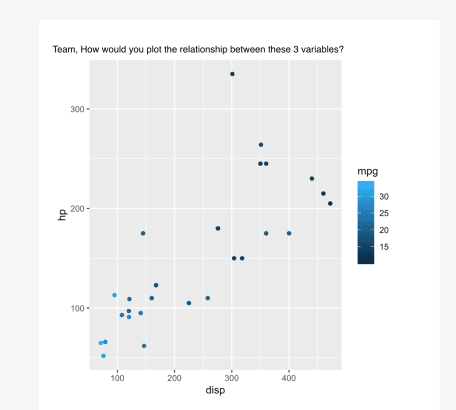
body = md(c(

"Team, How would you plot the relationship between these 3 variables?",

plot\_email

))

)



Visit the [documentation](https://rich-iannone.github.io/blastula/) to learn how to embed  
images, set email headers and footers, and even add call-to-action buttons.

**Sending Custom Emails with SMTP**

To send email, blastula includes functions to access SMTP servers such as GMail, Outlook, and Office365.

First, securely tell blastula about your SMTP server:

create\_smtp\_creds\_key(

id = "gmail",

user = "[user\_name@gmail.com](mailto:user_name@gmail.com)",

provider = "gmail"

)

Next, use the SMTP service to send your custom email:

email %>%

smtp\_send(

from = "[personal@email.net](mailto:personal@email.net)",

to = "[personal@email.net](mailto:personal@email.net)",

subject = "Testing the `smtp\_send()` function",

credentials = creds\_key(id = "gmail")

)

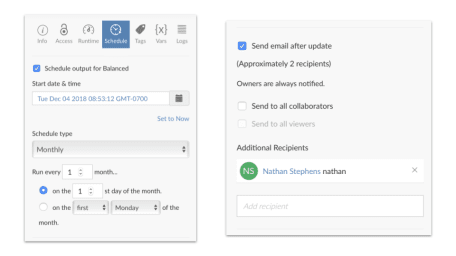
**Sending Custom Emails with RStudio Connect**

Organizations can use blastula in production on RStudio Connect. For instance, we  
use blastula to track critical services like our support ticket  
volume and our staff training schedules.

An easy way to get started is to access the RStudio Connect examples:

blastula::prepare\_rsc\_example\_files()

Publish the resulting R Markdown document to RStudio Connect, where it can be  
[scheduled for regular  
execution](https://docs.rstudio.com/connect/1.7.8/user/r-markdown-schedule.html)  
and distributed to stakeholders.



Blastula offers three additional functions to make it easier to create emails  
for RStudio Connect.

* render\_connect\_email automatically adds a footer to the email with useful links back to the content on RStudio Connect.
* attach\_connect\_email ensures RStudio Connect sends the custom email, and also makes it easy to customize the subject line, include additional email attachments, and optionally attach the report output.
* suppress\_scheduled\_email() allows you to skip sending the email. This pattern is very powerful. For example, reports can be run once a day, but only distributed if certain conditions are met.

Together, these three functions can be used to send proactive notifcations:

if (demand\_forecast > 1000) {

render\_connect\_email(input = "alert-supply-team-email.Rmd") %>%

attach\_connect\_email(

subject = sprintf("We need to prepare %d units!", demand\_forecast),

attach\_output = TRUE,

attachments = c("demand\_forecast\_data.csv")

)

} else {

suppress\_scheduled\_email()

}